## REMARKS

### Pending claims

Assuming entry of this amendment, claims 2-4, 6-9, 12-17, and 20-33 are pending, of which claims 21-33 are new and claims 12, 13, 21, 22, 27 and 30 are independent. Claim 21 is an amended version of previous claim 1.

### Specification Amendment

The change (other than the correction of a typographical error) requested in the specification relates to the definition of the virtual machine monitor. As stated in the last sentence, "[t]he general features of VMMs are known in the art and are therefore not discussed in detail here." Thus, this paragraph is summarizing known features of virtual machine monitors. One known feature of modern VMMs is that it is not necessary (although frequently implemented) for a VMM to virtualize all the hardware resources of the hardware platform on which it runs; rather, it may virtualize only a subset, or, indeed, a superset, of the resources, and these need not relate to the actual hardware present. The change to the specification is requested in order to reflect this known property; it adds no new matter, but rather simply better describes known VMMs.

### Claim Rejections

#### Rejections Under 35 U.S.C. 102

The Examiner rejected claims 1-4 and 7-20 under 35 U.S.C. 102(e) as being anticipated by US6,134,596 ("*Bolosky*").

*Bolosky* deals with the problem of scheduling access to a network on the part of several different data servers that cooperate to act as a single "continuous media file server" 50, but that have data files with different transmission rates. With reference to original claim 7, the Examiner wrote:

> … Bolosky teaches the system of claim 4 in which:
> the system resource is system machine memory (see col. 13 lines 35 - col. 14 lines 22, col. 19 lines 29-col. 20 lines 2 and col. 12 lines 36-51);
> the guest operating system allocates and deallocates physical memory to applications and drivers loaded within and connected to the guest

operating system (see col. 13 lines 35 -col. 14 lines 22, col. 19 lines 29-col. 20 lines 2 and col. 12 lines 36-51);

upon an increase in the resource quantity request for a specified one of the drivers, the guest operating system reserves a corresponding quantity of memory (see col. 13 lines 35 - col. 14 lines 22, col. 19 lines 29-col. 20 lines 2 and col. 12 lines 36-51);

upon a decrease in the resource quantity request for the specified one of the drivers, the guest operating system deallocates a corresponding quantity of physical memory (see col. 13 lines 35 - col. 14 lines 22, col. 19 lines 29 - col. 20 lines 2 and col. 12 lines 36-51).

The applicant respectfully disagrees with this assertion on several grounds:

The primary limited resource in *Bolosky's* system is network bandwidth (although disk bandwidth and buffer memory capacity are also mentioned). If network bandwidth were unlimited, then there would be no need to schedule network access among the different data servers 24 at all and the whole reason for *Bolosky's* scheduling mechanism would disappear since it would then not matter what transmission rates the various data files had. Among the many examples of instances where *Bolosky* makes this clear are (emphasis added):

> As represented in FIG. 9, the system's network scheduler ensures that there are sufficient network resources to output the multiple streams at multiple data rates. Here, the network scheduler schedules eight data streams of different data rates in a manner that avoids over-utilization of the **network bandwidth**. (col. 14, lines 23-27)

and

> The network scheduler ensures that the instantaneous **network bandwidth** never exceeds the maximum data rate of a server's network card, for all failure modes. (col. 15, lines 2-4)

In contrast, independent claims 12, 13, 21, 27, and 30 all make clear that the limited resource that is shared among the various guest systems in the embodiments of the invention defined in these claims is the hardware memory, which the various guest systems allocate and address as guest physical memory. The relationship between the concepts of hardware (machine) memory and guest physical memory is explained in the specification section entitled "Physical, machine and virtual memory" beginning on page 13.

Accordingly, as recited in claims 12, 21, 27 and 30, in the system implementations of the invention, a "memory reservation software module" is included in each guest system "for receiving a memory quantity request from the host system and for changing the allocation of the guest physical memory from within the respective guest system/OS according to the memory quantity request, thereby changing the amount of the hardware memory available for arbitrary use by the host system."

An analogous method step is recited in claim 13: "changing the allocation of the guest physical memory from within the guest OS in response to a memory quantity request issued by the host system, thereby changing the amount of the hardware memory available for arbitrary use by the host system."

No such memory reservation software module or corresponding method step is mentioned anywhere in *Bolosky*. *Bolosky* has no teaching of the concepts of guest physical memory and hardware memory and, accordingly, no mention of any mechanism for communicating any form of memory-related request between either scheduler 52, 54 shown in his controller 22 and any server 24. Moreover, there is no indication in *Bolosky* that any action on the part of any server would affect the availability of any form of resource to any form of "host system."

*Bolosky* also mentions disk bandwidth and buffer memory as being "main resources that are allotted to the data streams" (col. 4, lines 40-42). There no mention in *Bolosky* that disk bandwidth is a resource with changeable availability, however; accordingly, *Bolosky* provides no mechanism to change it. Moreover, disk bandwidth is generally a hardware characteristic that is unrelated to the novel teachings of the present applicant.

Concerning buffer memory 36, *Bolosky* fails to teach either a specific memory reservation software module or any direct communication to any server from any scheduler of any request to change the size of any buffer. Rather, *Bolosky* makes clear that adjustment of buffer size to accommodate a *file* (note: not memory) request is the task of each server, with no indication by the network scheduler of how a given buffer should be adjusted. See for example, *Bolosky's* "Buffer Memory Management" section beginning on col. 21, line 11. To illustrate this point further, consider what happens in *Bolosky* if current buffer availability is too small, as shown in Figures 17, 18 and 20:

As shown in Figure 17, if buffer capacity is exceeded (step 152), the new data stream is rejected (step 156);

As shown in Figure 18, if there is insufficient buffer capacity for a new stream (step 166), then a new search time must be set (step 172); and

As shown in Figure 20, if there is not a sufficient buffer resource in the local schedule (step 202), then the process is ended altogether.

In short, in contrast to the applicant's invention as claimed, if *Bolosky's* scheduler detects insufficient buffer memory in a given server, then he discloses no mechanism by which the scheduler can have it increased.

In the implementation of the invention defined in claim 22, the resource that is allocated among the various guest systems is the use of "<u>a plurality of cooperating hardware processors that are included in the host system and that are collocated in a single hardware platform</u>." In considering this multi-processor embodiment of the invention, the examiner wrote (with respect to earlier claim 11):

> *Bolosky* teaches the system of claim 1 where:
> the host system includes a plurality of processors (see col. 1 lines 24-37 and col. 11 lines 54-62); and
> the system resource is the plurality of processors, the resource quantity request indicating to the resource request means in each respective guest system a number of the plurality of processors to be reserved by each guest system thereby making the reserved processors available for reallocation by the host system (see col. 13 lines 35-col. 14 lines 22, col. 19 lines 29-col. 20 lines 2 and col. 12 lines 36-51).

The applicant respectfully disagrees. Although *Bolosky* states (col. 11, lines 54-57) that the "controller and data servers can be implemented, for example, using general purpose computers. Such computers include conventional components such as one or more data processors," there is no teaching anywhere in *Bolosky* that a given data server would be able to reserve or relinquish use of any processor that is not already part of its own hardware. Indeed, *Bolosky's* system would not benefit by doing so, since additional processors would not improve network bandwidth, disk bandwidth, or buffer memory capacity, which are the only resources *Bolosky* mentions (again, col. 4, lines 40-42).

In contrast, claim 22 recites that "each guest OS [is] provided with a processor reservation software module comprising computer-executable code for receiving from the host system a respective processor quantity request, which indicates a number of the plurality of processors to be reserved by each guest system, and for indicating to the guest OS a change in the number of processors reserved for use by the respective guest system, thereby making the reserved processors available for arbitrary use by the host system." No such features are disclosed by or even hinted at in *Bolosky*.

Claims 22, 27, and 30 recite another feature of certain embodiments of the invention that is not disclosed in *Bolosky*, namely, that either multiple hardware processors, the guest systems (virtual computers or otherwise), or both, are within a single, physical machine. Thus,

Claim 22: "a plurality of cooperating hardware processors that are included in the host system and that are collocated in a single hardware platform" which can be "reserved for use by the respective guest system"

Claim 27: "the guest physical address spaces of all the virtual computers being mapped as portions of the hardware memory address space" and "the virtual computers all being executable on the same hardware processor(s), which are collocated in a single hardware platform"

Claim 30: "the guest systems are all executable on the same hardware processor(s), which are collocated in a single hardware platform"

The invention as defined in these claims would not be possible if the host and guest systems were each, and separately "implemented, for example, using general purpose computers" (col. 11, lines 54-55) as taught in *Bolosky*.

## Rejections Under 35 U.S.C. 103

The Examiner rejected claims 5 and 6 under 35 U.S.C. 103(a) as being unpatentable over *Bolosky* in view of US6,247,081 ("*Murata*"). In particular, the Examiner wrote (emphasis added):

> *Murata* teaches a file management system that controls access to storage devices using a virtual machine that controls scheduling and virtual memory management (see col. 4 lines 7-30).
>
> It would have been obvious for one of the ordinary skill in the art at the time of the invention to modify the guest operating system of *Bolosky* by incorporating virtual machines as taught by *Murata* because doing so would allow the user to monitor the resource schedule using a web interface, **Java and DB2**.
>
> ... *Murata* virtual machine monitor forming an interface between the resource scheduler and each respective virtual machine (see col. 4 lines 7-30).

The embodiments of the applicant's invention in which the guest system(s) are virtual computer(s) are defined in claims 12, 13, 21, and 27. The concept of a "virtual machine" as used in this invention is shown in Figure 1 and is discussed in the specification, page 12, lines 12-19:

> As is well known in the field of computer science, a virtual machine (VM) is a software abstraction -- a "virtualization" -- of an actual physical computer system. As such, each VM 300 will typically include a virtual CPU 310 (VPROC), a guest operating system 320 (which may simply be a copy of a conventional OS), virtual system memory 330 (VMEM), a virtual disk 340 (VDISK), virtual peripheral devices350 (VDEVICES) and drivers 322 (VDRIVERS) for handling the virtual devices 350, all of which are implemented in software to emulate the corresponding components of an actual computer.

There are two different software entities that are referred to in the field of computer science as a "virtual machine": 1) a runtime system that interprets or compiles a platform-independent instruction set, such as the Java bytecodes defined for the Java programming language, or the P-code instructions defined for the UCSD Pascal language; and 2) a virtual machine as found in the products of VMware, Inc., of Palo Alto, California, which is the type described in the portion of the specification cited above. Claims 12, 13, 21 and 27 recite specific software features that define the intended virtual computers as being of the second type. Note that, in the field of computer virtualization, the virtual machine and its related supporting virtual machine monitor may be separate software entities, but are often viewed as a forming a single virtual "computer." The virtual machine *monitor* used to support such a virtual machine in the applicant's invention is described on page 13, first paragraph.

*Murata* uses the word "virtual" only three times, all in a portion of the paragraph cited by the Examiner (emphasis added), col. 4, lines 7-24:

> As illustrated, the operating system services 120 includes multiple subsystems: a file management subsystem 130, a network subsystem 132, a services 134, a **virtual** machine manager subsystem 136, drivers 138, an object manager 140, a security reference monitor 142, and a process manager 144. The file management subsystem 130 controls accesses to the storage devices, including floppy drives, CD-ROM drives, hard drives, etc. The network subsystem 132 controls the networking of a computer system to one or more other computer systems. A binding engine 133 within the network subsystem 132 controls the binding of various network drivers to one another in accordance with the multi-layered driver architecture, as discussed in more detail below. The services 134 provides control over hardware profiles for the operating system as well as additional "miscellaneous" operating system finctions. The **virtual** machine manager subsystem 136 controls task scheduling and **virtual** memory management.

As *Murata's* Figure 1 shows, the virtual machine *manager* 136 is a software entity entirely within the operating system 110, along with such other entities as device drivers 138 and a file management subsystem 130. No virtual machines are illustrated or described at all. Consequently, to the extent *Murata* supports any form of virtual machine, then it is of a type similar to a Java virtual machine.

That *Murata's* virtual machine manager 136 "controls task scheduling and virtual memory management" is likewise no indication that *Murata* discloses a virtual machine in the sense of the applicant's invention. As is well known in the art of computer science, for example, from such standard texts as "Computer Organization and Design: The Hardware/ Software Interface," by David A. Patterson and John L. Hennessy, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1994, pp. 579-603 (chapter 7.4 "Virtual Memory"), "virtual memory" is present in almost every modern computer, and is unrelated to the concept of "virtual" as used in the term "virtual machine," of either type.

Consequently, *Murata* does *not* disclose any claimed feature of the invention that, together with the teachings of *Bolosky*, would render claims 12, 13, 21, and 27 obvious.

The applicant concedes the existence of many references that disclose virtual machines as found in this invention, but this is irrelevant to the question of the novelty and non-obviousness of the present invention: This invention does not relate to virtual machines as such, but rather to a system and method for dynamically changing the amount of a system resource, for example, memory or processors, that are reserved by a virtual machine or other guest system. Even if the data servers 24 in *Bolosky were* virtual machines, however, such a hypothetical system would still not display the novel features of the different embodiments of the applicant's invention as defined in the independent claims: As explained above, *Bolosky* does not disclose these features, and known virtual machines do not have them, so neither would a combination.

## Claim amendments in general

Most of the requested amendments to the claims, which are also incorporated in the new claims, relate to clarifications of the concepts of two different aspects of memory, namely guest physical memory and hardware memory, and to the nature and operation of the resource reservation module in the guest systems. Where appropriate, the general "guest system" recited in some independent claims is explicitly referred to in others as a virtual computer and the resource is explicitly stated to be hardware memory. The claims that are dependent from the new independent claims 22, 27 and 30 are in general "copies" (modified where necessary for consistency and to ensure antecedent basis) of the claims dependent on independent claim 21, which replaces original claim 1. The requested changes to the claims not only recite the various embodiments of the invention so as to more clearly distinguish the respective embodiments from the cited prior art, but also to make the structure of the embodiments more readily understandable.

**Conclusion**

The various embodiments of the applicant's invention as defined in the corresponding independent claims recite features that are not found at all in either of the cited references, whether the references are viewed independently or in combination. As such, the independent claims should now be allowable over the cited prior art. The various dependent claims of course simply add additional limitations and should therefore be allowable along with their respective independent base claims.

**Change of Attorney Name**

Please note the enclosed letter concerning the change of the attorney's name from "Slusher" to "Pearce."

Date: 16 April 2004

Respectfully submitted,

Jeffrey Pearce

34825 Sultan-Startup Rd.
Sultan, WA 98294
Phone & fax: (360) 793-6687

Jeffrey Pearce
Reg. No. 34,729
Attorney for the Applicant